

OMS GUI Admin Tools Features

Supplementary Documentation

Abstract

The Admin Tool feature of the OMS GUI allows a privileged operator to configure permissions for particular elements on pages of the OMS GUI. Using this feature, an administrator can limit access to or execution of a configuration parameter or action. This is based on setting configuration parameters to “r” (read-only) or “rw” (read-write) and actions as yes/no (internally they are still r/rw). This manual will outline and explain how this feature works.

The ADMIN Account

In order for an operator to obtain access to the Admin Tools function, he/she must be logged as a configured administrator. Any one operator can be configured to have admin privileges in the configuration file **EcOmGui.CFG**:

```
ADMIN_OPERATOR=jsmith
```

The operator “jsmith” above must already exist as an authorized “readWrite” operator in the password authentication file located under /usr/ecs/CUSTOM/<MODE>/data/OMS/. This file has a naming convention of Password_OMS_<MODE>.txt. So for OPS mode, the complete path and name of the file would be:

```
/usr/ecs/OPS/CUSTOM/data/OMS/Password_OMS_OPS.txt
```

This file contains the list of authorized operators and their permissions. One of these operator IDs can be set to ADMIN_OPERATOR as indicated above, or a new operator can be added to this file with the appropriate application (separate documentation is provided on GUI security and how to add/remove/update operators).

When logged in as an ADMIN operator, you will see a warning in the navigation frame and an additional menu item called “Admin Tools”. Any non-admin operator will not see this menu item and therefore does not have access to it.

Operational Overview

First, log in to the OMS GUI as an ADMIN operator, and then expand the **Admin Tools** menu in the navigation frame. Under this you will see several menu items that correspond to the OMS GUI pages, for example, **Server/Database Parameters**, which corresponds to the Server/Database pages under OM Configuration.

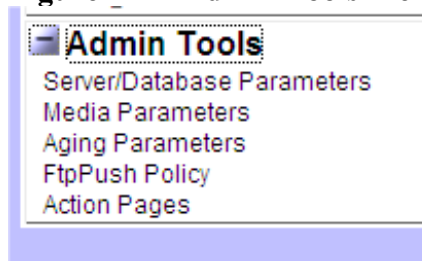
The idea is to create permissions for each operator for each page. A drop-down list will appear at the top of the page, containing the list of all “readWrite” operators (“read” operators are not eligible since by definition they can only view the GUI and not actually operate it).

Figure 1-1: OMS GUI running in ADMIN mode



Figure 1-1 shows that the Admin Tools menu is available when logged in as an ADMIN operator. **Figure 1-2** shows the detail of the menu items available for this feature.

Figure 1-2 – Admin Tools Menu Items



Setting Operator Permissions

If no permissions have previously been set for an operator, a warning will be displayed saying that no profile exists for the selected operator. This means the configuration file contains no data for this operator and for this page (even though permissions may have been set for that operator on another page).

Here are step-by-step instructions for setting operator permissions:

1. Select the page for which permissions will be set, e.g., “Server/Data Parameters”.
2. Select the operator ID from the drop-down list at the top of the page:

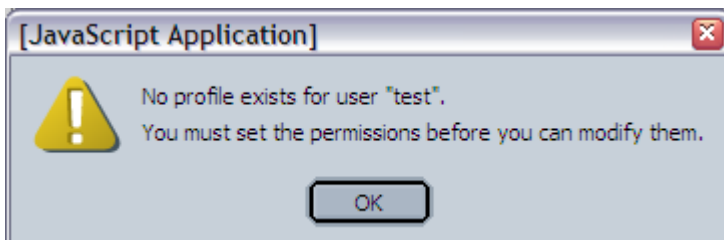


Set Permissions for **jpino**


3. The permissions will automatically be loaded into the checkboxes for that operator:

OmServer	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
mail	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
ust be	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
C Personnel	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
f concurrent	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
PDS	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
f concurrent	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
SDSRV	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
f concurrent	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
PDS	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
to wait to	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
s before	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
Completed	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
re	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Completed	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
intained	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
ver of	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
uest may	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
ver of	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
uest may	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>
ecifies	r <input checked="" type="checkbox"/> rw <input type="checkbox"/>

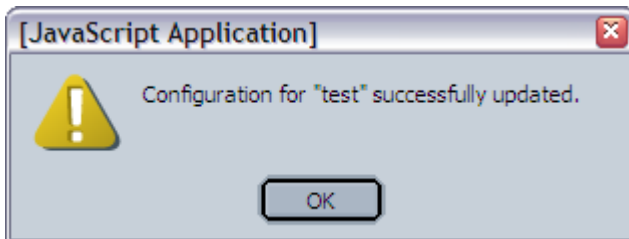
4. If no permissions have ever been set for that operator, a warning message will be displayed:



5. You can either check each box manually or check the bold “r” or “rw” which will check all boxes to “r” or “rw” respectively:

Description	Set Permissions for test 
	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Name used by OmServer for DORRAN Email Notifications must be updated by EDC Personnel	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Max Number of concurrent submissions to PDS	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Max Number of concurrent submissions to SDSRV	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Max Number of concurrent submissions to PDS	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>
Amount of time to wait to kill child process before retrying action	r <input type="checkbox"/> rw <input checked="" type="checkbox"/>

- Once you are done setting all the permissions, click “Apply Changes” next to the drop-down list.
- The page will reload. You should now receive a “success” message for updating the operator permissions:



- Click OK to continue. The page will be set back to its original state (i.e., no operator selected).
- Follow these same procedures for each successive page. Note that even if a “profile” exists for a user on one page, it may not exist for that user on another page.

NOTE: If no permissions are explicitly set for an operator, the default is that all permissions are set to “rw”, i.e., there are no restrictions for any parameter or action.

Once the permissions have been set for an operator, they can be modified by following similar procedures as above. The only way to “unset” the profile for an operator is to manually remove references to it in the configuration file or to remove the file (in which case *all* operator permission settings will be lost).

Operations Concept and Scenarios

For test engineers and developers, the following information may prove useful. The scenario for setting and retrieving operator permissions is as follows:

1. Operator clicks on a page link
2. The request is sent to the web server, whereupon the Perl code reads the file `EcOmGuiUserPerm.CFG`, which is actually a Perl module containing multidimensional hashes (this is faster and provides better performance than a conventional configuration file because it is in the native Perl format and requires no parsing).
3. The request is sent back to the client browser, where a JavaScript multidimensional hash is dynamically created in the new web page. All data for all operators is thus loaded into the client-side memory (for faster performance).
4. The operator selects an operator ID
5. The JavaScript code populates the checkboxes with the appropriate permission settings (or doesn't if no profile exists for the operator).
6. The operator submits the page and the data is sent back to the web server
7. The Perl code converts the data so it can be written to `EcOmGuiUserPerm.CFG` as native Perl hashes. Any existing entries for that operator and that page are first removed and placed into a write cache. The modified cache is then written back to the configuration file
8. The page is reloaded on the client side, and actually loads twice – once to alert the operator that the change has taken place, and again to load the hashes into memory.